TPF Users Group Spring 2005

TPF Toolkit Education Session

From here to there

Mary Huang

AIM Enterprise Platform Software

IBM z/Transaction Processing Facility Enterprise Edition 1.1.0

© IBM Corporation 2005

Any references to future plans are for planning purposes only. IBM reserves the right to change those plans at its discretion. Any reliance on such a disclosure is solely at your own risk. IBM makes no commitment to provide additional information in the future.
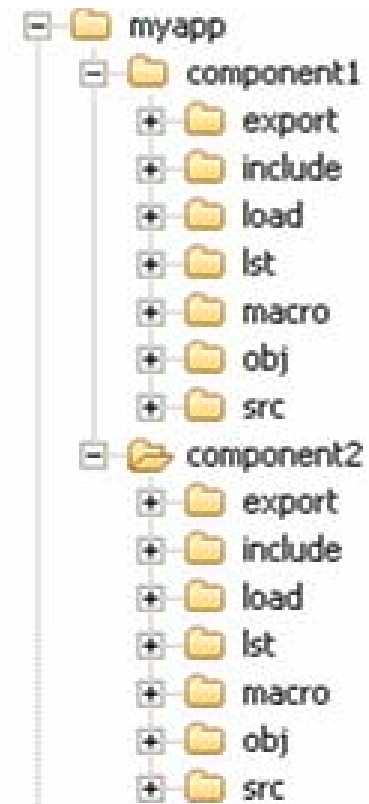
Agenda

- Overview of migration from TPF 4.1 to z/TPF

- Overview of MakeTPF

- How TPF Toolkit can help you

- Single Source Tools

Overview of migration

- Install TPF Toolkit

- Move source from PDS to HFS

- Move to MakeTPF build

- Convert to Single Source
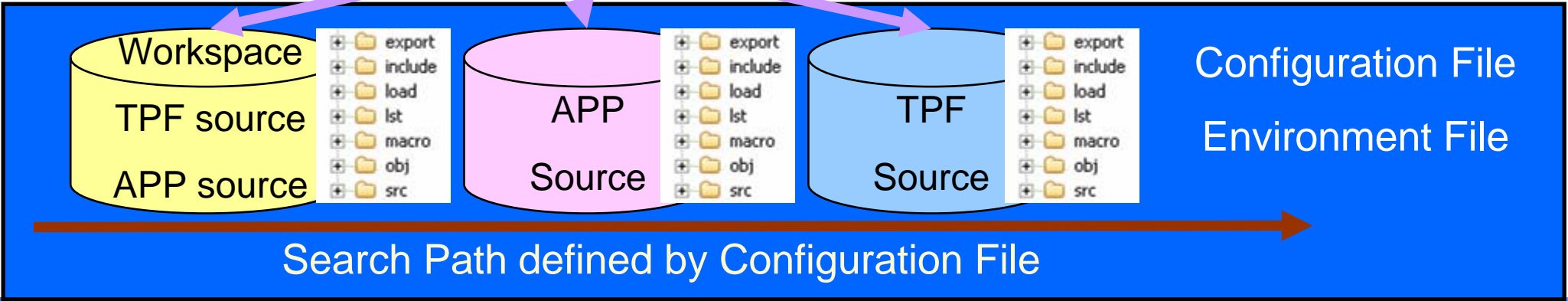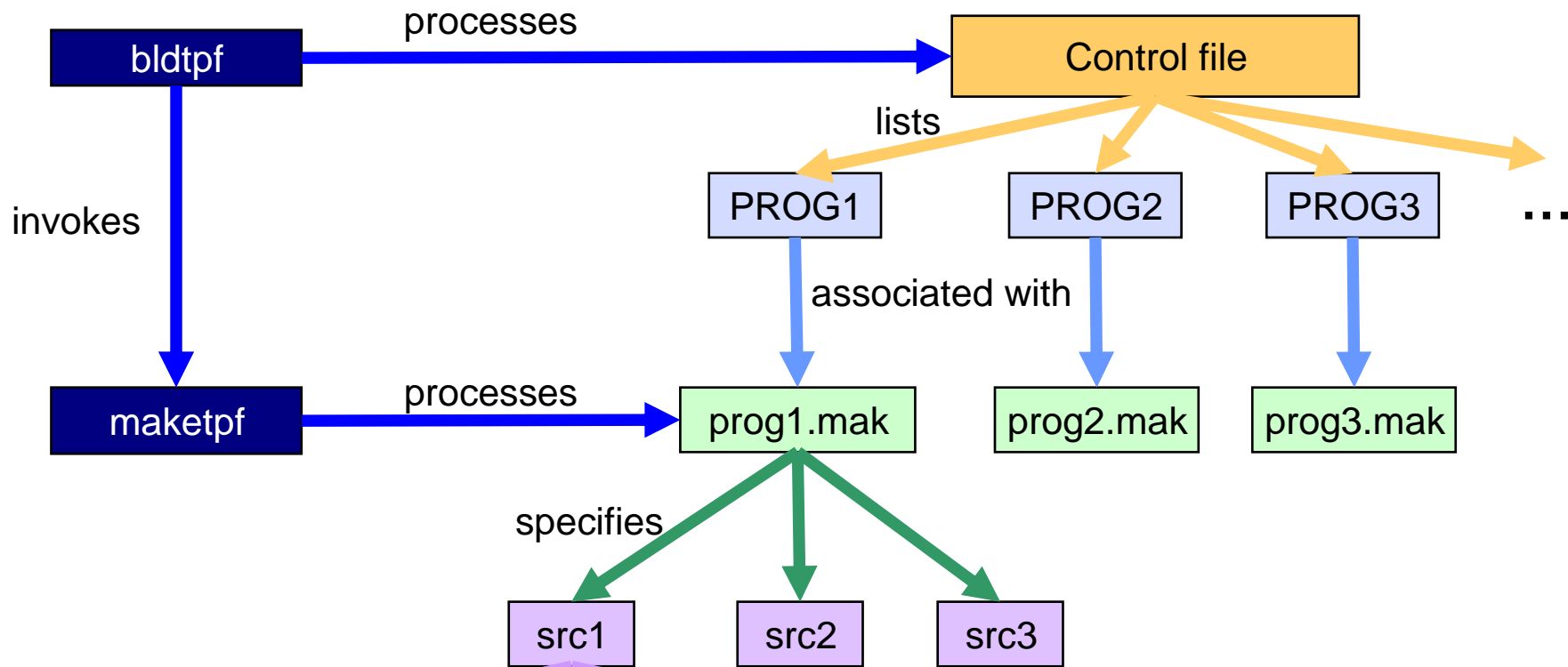
Move source from PDS to HFS

- z/TPF development is Linux-based
  - no more PDS
  - moving to USS/HFS today eases migration to the Linux-based environment
  - can re-use same directory structure in USS and Linux

- Define HFS directory structure for TPF 4.1 applications
  - root directory for each application
  - multiple application directories, divided by function
  - use sub-directories to set up logical groupings

```
myapp
  component1
    export
    include
    load
    lst
    macro
    obj
    src
  component2
    export
    include
    load
    lst
    macro
    obj
    src
```

Move to MakeTPF based build

- Recommended build tool set for z/TPF

- Common tool set for system programs and application programs
  – tools for assembling, compiling and linking both TPF system and application programs

- Same tools and makefiles for production and development builds

- Is available on both Linux and USS for both online and offline program builds

- TPF Toolkit supports MakeTPF based builds for TPF 4.1 and z/TPF

- Minimal makefile knowledge required

Overview of MakeTPF

bldtpf —— processes ——▸ Control file

bldtpf —— invokes ——▸ maketpf

Control file —— lists ——▸ PROG1   PROG2   PROG3   ...

PROG1 —— associated with ——▸ prog1.mak

maketpf —— processes ——▸ prog1.mak

PROG2 ——▸ prog2.mak

PROG3 ——▸ prog3.mak

prog1.mak —— specifies ——▸ src1   src2   src3

Workspace
TPF source
APP source

APP
Source

TPF
Source

export
include
load
lst
macro
obj
src

Configuration File

Environment File

Search Path defined by Configuration File

Overview of MakeTPF

- Files used by MakeTPF
  - Makefile
  - Configuration File
  - Environment File
  - Rules File
  - Control File

Overview of MakeTPF – Rules File

- Defines assemble, compile and link rules

- Defines default assemble, compile and link options

C Compile Rule

```
$(C_OBJS) : %.o : %.c $(DEPENDENCIES)
    @ $(maketpfmsg) "$(MTPF2001I)"
    $(CC) $(CFLAGS) -o '$@' -c '$<'
    $(MV) '$@' '$(OBJDIR)/$@'
```

CXX Compile Rule

```
$(CXX_OBJS_CPP) : %.o : %.cpp $(DEPENDENCIES)
    @ $(maketpfmsg) "$(MTPF2001I)"
    _CXX_CXXSUFFIX=cpp  _CXX_CCMODE=1 $(CXX) -o $@ -c $< $(CXXFLAGS)
    $(MV) '$@' $(OBJDIR)/$@
```

Overview of MakeTPF – Configuration File

- Defines the working build space

- List of root directories to include in the build
  - ordered list of directories dictates search order
  - first root directory used for outputs

- System and sub-system information

- Build options overrides
  - overrides for default assemble, compile, link options

- Other options
  - keep listings
  - run post process utilities

MakeTPF Config file

TPF source directories

TPF_ROOT := /u/userid/wrkspc/defect001/tpfsrc

TPF_ROOT += /u/tpf41/rlse

Application source directories

APPL_ROOT := /u/userid/wrkspc/defect001/appsrc
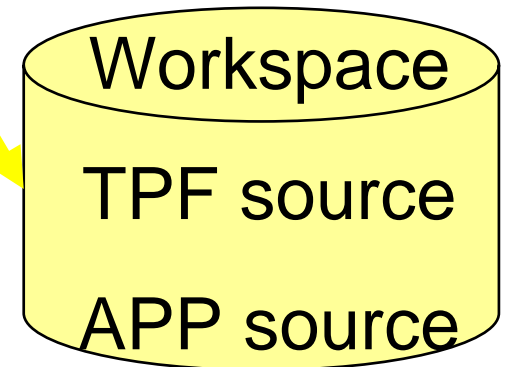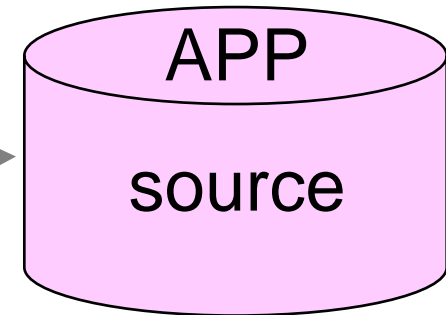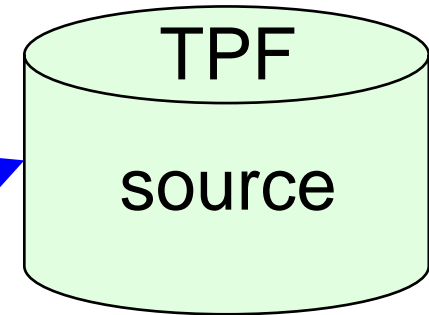
APPL_ROOT += /appsrc/cur

TPF System name

TPF_BSS_NAME := BSS
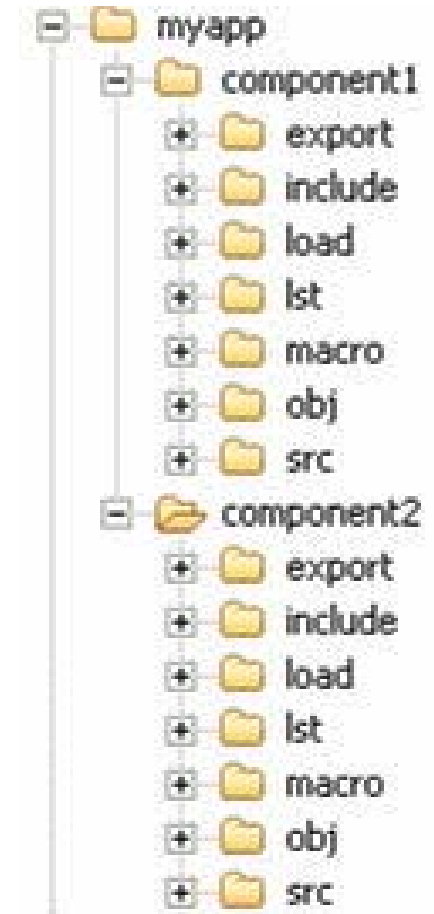
TPF_SS_NAME :=

Build options

ASMFLAGS_USER := -k

TPF_VERIFY_LINK_REFS := YES

TPF source

APP source

Workspace

TPF source

APP source

Search Path

Overview of MakeTPF – Environment File

- **Defines the directory structure under the root location**
  - logical grouping of source and output files
  - supports both flat and multi-level deep directory structures

- **Source and output directory information**

```
myapp
   component1
      export
      include
      load
      lst
      macro
      obj
      src
   component2
      export
      include
      load
      lst
      macro
      obj
      src
```

Environment File

Environment for TPF BASE Real Time Source and Output

ROOTLOADDIRS := $(foreach d,$(TPF_ROOT),$d/load)

ROOTEXPDIRS := $(foreach d,$(TPF_ROOT),$d/export)

ROOTOBJDIRS := $(foreach d,$(TPF_ROOT),$d/obj)

ROOTLSTDIRS := $(foreach d,$(TPF_ROOT),$d/lst)

ROOTINCDIRS := $(foreach d,$(TPF_ROOT),$d/include)

ROOTMACDIRS := $(foreach d,$(TPF_ROOT),$d/macro)
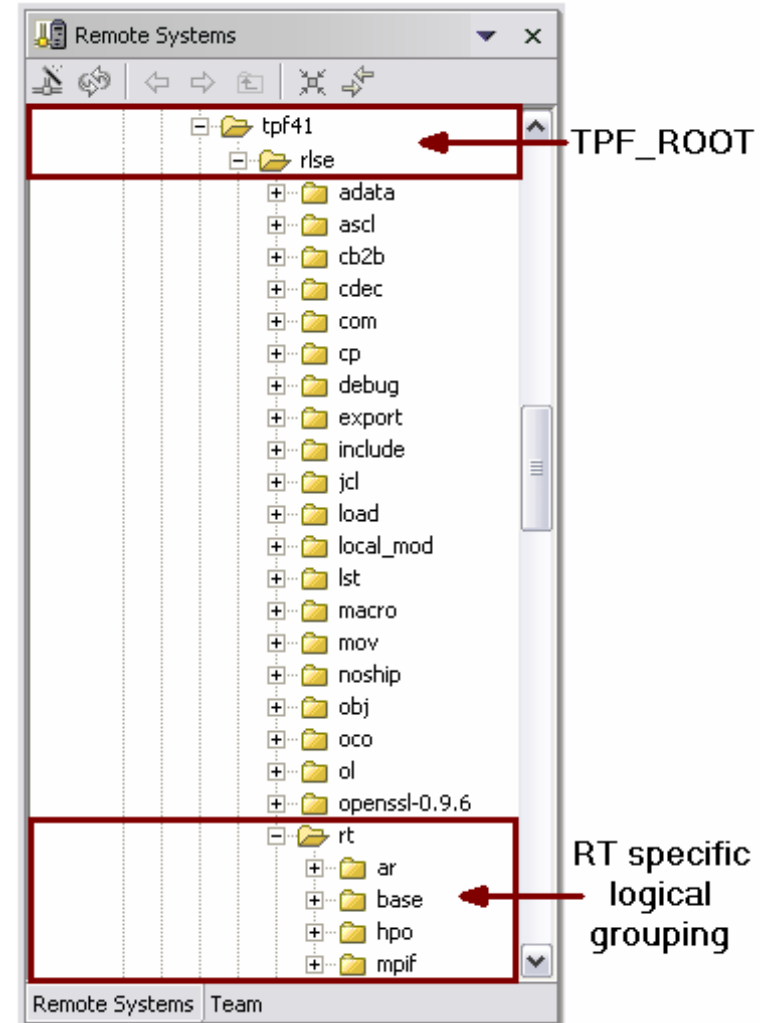
ROOTCPYDIRS := $(foreach d,$(TPF_ROOT),$d/**rt/base**)

ROOTASMDIRS := $(foreach d,$(TPF_ROOT),$d/**rt/base**)
ROOTASMDIRS += $(foreach d,$(TPF_ROOT),$d/**rt/mpif**)

ROOTCDIRS := $(foreach d,$(TPF_ROOT),$d/**rt/base**)
ROOTCDIRS += $(foreach d,$(TPF_ROOT),$d/**rt/ar**)
ROOTCXXDIRS := $(foreach d,$(TPF_ROOT),$d/**rt/base**)

Remote Systems

tpf41 ← TPF_ROOT
rlse
adata
ascl
cb2b
cdec
com
cp
debug
export
include
jcl
load
local_mod
lst
macro
mov
noship
obj
oco
ol
openssl-0.9.6
rt ← RT specific logical grouping
ar
base
hpo
mpif

Remote Systems | Team

Overview of MakeTPF - Makefile

- Program information
  - only one primary target program
  - name and type of the program

- Lists source files required to build the program
  - use base name of source files only

- Environment files
  - maps to sub-directory structure under root directories
  - dictates where to look for referenced files

- Special build options (to override default options)

- Includes a common maketpf.rules file

- Wizards in TPF Toolkit to generate a makefile

Program information

APP := ZDX2

APP_TYPE := DLL

Source segments

CXX_SRC := idFunctions.cpp

ASM_SRC := qt00.asm

Environments for this program

maketpf_env := app

maketpf_env += base_rt

Override options

LDFLAGS_ZDX2 := -M

**Include statement**

include maketpf.rules

Overview of MakeTPF – Control File

- **Controls the build**
  - List programs to build
  - defines build order
  - central point of information regarding all programs
  - used to set up for project builds
  - processed by bldtpf tool

- **Each entry has 32 columns (semi-colon delimited list)**
  - name of program to build
  - program type
  - makefile to use to build the program
  - system load attributes
  - etc…

Program Name	Program Type	Makefile Name	Num of build passes	System Allocation	OBJ shippable	Function switch	...
PROG1	APP	prog1.mak	1	BSS	OBJ	TPF_SBALL	...
PROG2	CIMR	prog2.mak	2	ALL	NOOBJ	TPF_SBALL	...

bldtpf in action…
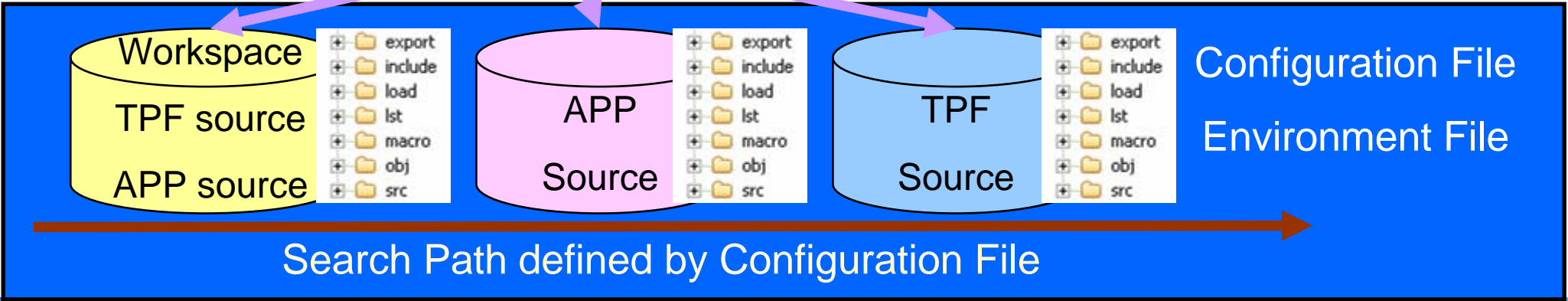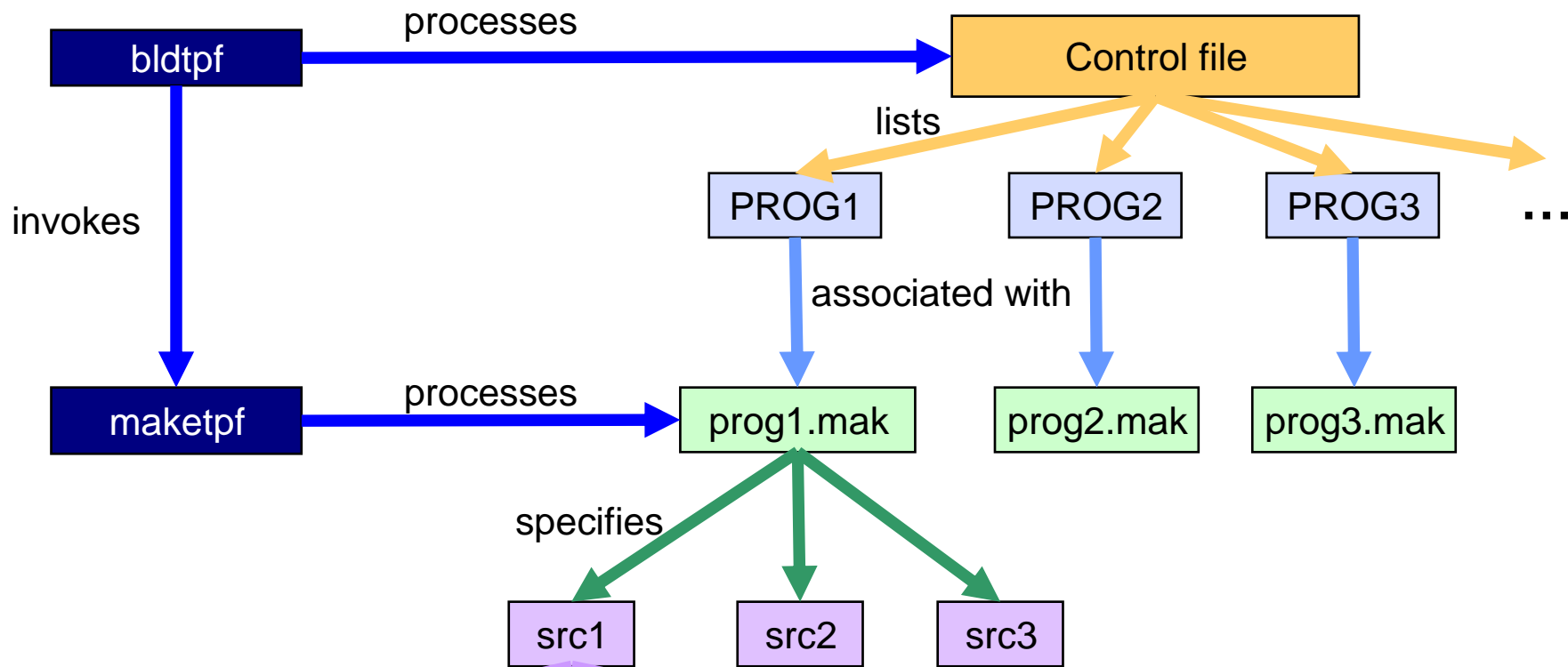
Converting to MakeTPF based builds

- **Move from PDS to HFS**
  - consider defining a directory that you can re-use for z/TPF

- **Understand MakeTPF**
  - MakeTPF education session at last TPFUG – download education material
  - download MakeTPF sample package
    http://www.ibm.com/software/htp/tpf/download/maketpf.htm

- **Create MakeTPF environment files**
  - map HFS directory structure to environment files
  - can re-use environment files in z/TPF if same HFS directory structure is maintained

- **Create Rules file for the build**
  - define default build options for compile, assemblies, links
  - compile, assemble and link rules for your programs

Converting to MakeTPF based builds

- Convert build scripts to makefiles (maketpf.bsc.convert)
  - tools to assist converting DLM, DLL and LLM build scripts to MakeTPF makefiles
  - run tool from command line as well as from TPF Toolkit
  - demo


```
maketpf.bsc.convert bsc [-i appl_roothfs] [-e env_list] [-t tpf_roothfs] [-o
 dir]


    bsc is the name of the build script to process
        a full or relative pathname must be specified.
     -i defines the source hfs to use to locate application source files.
     -e defines a maketpf_env name to be included in the makefile.
        if defined, the envs specified are included in the order specified.
     -v defines the version code for the application source files
        to be used in addition to the TPF version codes
        (40, 41, R0, H0, M0) when searching.
     -t defines the source hfs to use to locate TPF source files.
        by default /u/tpf41/intg is used.
     -o defines the output directory to write the makefile.
        by default $PWD is used.
```

Converting to MakeTPF based builds

- **Use MakeTPF in TPF Toolkit**
  - TPF Toolkit supports maketpf based builds for TPF 4.1
  - wizard to generate makefiles
  - wizard to generate configuration files
  - wizard to generate control files
  - build projects using bldtpf/maketpf
  - demo

  - demo with custom action

Convert to Single Source

- Make changes to 4.1 application to make them compatible for z/TPF

- Can maintain single copy of application source
  - build the same application source for TPF 4.1 or z/TPF without any conditional code
  - make changes to applications now while they are running in TPF 4.1

- Conversion tool in TPF Toolkit

- Details by Andrea

TPF Toolkit and z/TPF

- Connectivity to USS and Linux for TPF 4.1 and z/TPF development

- TPF Toolkit V3 supports MakeTPF for TPF 4.1 and z/TPF
  - same interface
  - isolate application developers from implementation
  - demo

- Minimal change required to TPF 4.1 applications

Where to find more information

- z/TPF Migration Portal
  http://www.ibm.com/software/htp/tpf/pages/ztpfmigration.htm

- TPF Support, Services and Education

- Education Session material

Trademarks

IBM , zSeries and Websphere are trademarks of International Business Machines Corporation in the United States, other countries, or both.

Java and all Java-related trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc., in the United States and other countries.

Microsoft and Windows are registered trademarks of Microsoft Corporation.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both

* All other products may be trademarks or registered trademarks of their respective companies.

Notes

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States and Canada. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

This presentation and the claims outlined in it were reviewed for compliance with US law. Adaptations of these claims for use in other geographies must be reviewed by the local country counsel for compliance with local laws.